Experience with a Computer-Assisted Formal Programming Examination

John English
University of Brighton
Rationale

- Increasing student numbers
  - increasing marking burden
- Fewer assessments per module
  - less practice for students
- Plagiarism
  - students plagiarise when they can't cope
Approaches

- "Little but often" assessment
  - large amount of assessment requires automated marking
  - plagiarism problems
Approaches

• Tailored exercises
  – randomly selected and/or parameterised for each student
  – difficult to produce parameterised programming problems
Approaches

- Formal examinations
  - written exams are a poor test of programming ability
  - practical programming exams require access to compilers and other tools
Our solution

- Online programming exam for first-year students
  - supports a variety of question types
- Allows for special needs students
- Addresses health & safety issues
- Semi-automated marking performed offline after completion of exam
Question types

- Multiple choice
- Multiple select
- Arrange the answers
  - all these can be marked completely automatically
- Free text
  - needs manual moderation of automated marking
System design

- Rubric page
- CGI script (randomisation and rendering)
- Exam (XML)
System design

Rubric page

CGI script (randomisation and rendering)

Exam (XML)

Rendered exam

Student logfile
System design

- Rubric page
- Rendered exam
- Exam logging (applet and JavaScript)
- CGI script (randomisation and rendering)
- Exam (XML)
- Exam logging (server)
- Student logfiles
Management issues

- Students fill in and sign the normal paper exam form
  - name, student number, place number
  - online rubric form requires student number only
- Time allowances can be adjusted on a per-user basis
  - e.g. for dyslexic students
Security issues

- Authenticating server used to require login
- Exam can be made available to selected users
  - we use special accounts with no email or printing facilities
- Applet sends form content to server every 30 seconds
Health & safety

• We cannot require students to use a computer for 2 or more hours without a break
  – server allows disconnect and reconnect
  – disconnected time is deducted from a "permitted break" allowance
Exam format

- 2.5 hours + 30 minutes break time
- Section A: 10 multiple choice questions, 3 marks each (30%)
- Section B: 5 short practical questions, 6 marks each (30%)
- Section C: 2 longer practical questions, 20 marks each (40%)
Experience

• Used in 2001 and 2002
  – 2001: 64 students
  – 2002: 59 students

• Marking time drastically reduced
  – approx. 15 minutes for automatic marking
  – approx. 2 hours for manual moderation
Results (2001)
Conclusions

- The system tests practical programming ability
  - students can use the compiler to check their answers
- Problems arise from not following the instructions!
  - manual moderation needed if solutions don't compile, or if no test cases run successfully
Conclusions

- Marking workload drastically reduced
  - down from 10-15 hours (estimated) to about 2 hours
- Large corpus of data on student activity throughout the exam period
  - still to be analysed...
- Positive reaction from students
  - felt it worked well & was fair
Any Questions?